Report from Dagstuhl Seminar 13411

Deduction and Arithmetic

Edited by

Nikolaj Bjørner[1], Reiner Hähnle[2], Tobias Nipkow[3], and Christoph Weidenbach[4]

1 **Microsoft Research, Redmond, US,** nbjorner@microsoft.com
2 **Technische Universität Darmstadt, DE,** haehnle@cs.tu-darmstadt.de
3 **Technische Universität München, DE,** nipkow@in.tum.de
4 **Max-Planck-Institut für Informatik, Saarbrücken, DE,** weidenbach@mpii.de

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 13411 "Deduction and Arithmetic". The aim of this seminar was to bring together researchers working in deduction and fields related to arithmetic constraint solving. Current research in deduction can be categorized in three main strands: SMT solvers, automated first-order provers, and interactive provers. Although dealing with arithmetic has been in focus of all three for some years, there is still need of much better support of arithmetic. Reasong about arithmetic will stay at the center of attention in all three main approaches to automated deduction during the coming five to ten years. The seminar was an important event for the subcommunities involved that made it possible to communicate with each other so as to avoid duplicate effort and to exploit synergies. It succeeded also in identifying a number of important trends and open problems.

Seminar 6.–11. October, 2013 – www.dagstuhl.de/13411
1998 ACM Subject Classification D.2.4 Software/Program Verification, F.3.1 Specifying and Verifying and Reasoning about Programs, F.4.1 Mathematical Logic, G.1.2 Approximation, G.1.3 Numerical Linear Algebra, G.1.6 Optimization, I.2.3 Deduction and Theorem Proving
Keywords and phrases Automated Deduction, Program Verification, Arithmetic Constraint Solving
Digital Object Identifier 10.4230/DagRep.3.10.1
Edited in cooperation with Jasmin Christian Blanchette

1 Executive Summary

Nikolaj Bjørner
Reiner Hähnle
Tobias Nipkow
Christoph Weidenbach

Arithmetic plays a fundamental role in deduction. Logical constraints over arithmetical properties occur frequently in classical theorems in mathematics, as well as in program analysis and verification. The first automatic theorem prover was an implementation of Presburger Arithmetic in 1954. With the availability of powerful predicate calculus proof procedures some years later, arithmetic would be relegated to the sidelines. Interest in arithmetic revived in the 1980s with the advent of powerful interactive theorem provers that needed and supported arithmetic for their applications. The need for efficient computer aided deduction with support for arithmetic in the area of program analysis and verification recently gave birth to a new technology, so called SMT solvers.

Thus we have three strands of automated deduction: SMT solvers, automated first-order provers, and interactive provers in need of (more) arithmetic.

SMT SMT (satisfiability modulo theories) solvers distinguish themselves by integrating built-in support for a combination of theories, including prominently the theory of arithmetic. Most often handling arithmetic formulas in isolation is not sufficient. Applications typically use a non-disjoint combination of arithmetic and other theory reasoning. SMT solvers nowadays handle quantifier-free arithmetic well, but are not directly equipped to solve arithmetical formulas with quantifiers. Recent progress on building in quantifier-elimination procedures for linear and non-linear arithmetic have made practical integration of such richer arithmetic deduction viable.

ATP Research in first-order logic theorem proving used to concentrate on efficient calculi in general and the integration of equational theories in particular. It is obvious that further integration of "richer" arithmetic theories into first-order logic should be done by rather a combination approach than an integration approach. One major challenge of combining first-order logic calculi with arithmetic procedures is that of compactness/completeness and termination. While Boolean combinations of ground atoms, as they are considered by SMT solvers typically do not cause trouble with respect to those challenges, combining first-order clauses with an arithmetic theory can never result in a compact/complete/terminating calculus, in general. The actual combination typically requires the solution of purely arithmetic problems in order to establish valid inferences and simplifications. These problems are of a specific nature in that the form of the arithmetic formulas and the way they need to be tested require specific variants of the known arithmetic procedures.

ITP Interactive theorem provers initially came with built-in decision procedures for quantifier-free linear arithmetic. More foundational systems then developed new techniques to implement these decision procedures by reducing them to pure logic, trading efficiency for guaranteed correctness. Aspects of arithmetic reasoning are present in deductive software verification systems: interactive systems combine a number of automatic arithmetic reasoning methods and control them with heuristics that are specific for verification. A challenging application of interactive proof and arithmetic is the Flyspeck project, an effort to formalize Hales's proof of the Kepler conjecture in an interactive theorem prover.

The Dagstuhl seminar was a timely event that brought together experts in the above subareas of deduction, and in reasoning about arithmetic, to exchange experiences and insights. The research questions pursued and answered included:

- Which arithmetic problems are best solved with which approach?
- How to handle very complex numeric representations such as the IEEE floating-point standard with a high degree of automation?
- Arithmetic in combination with other theories results easily in languages with a very complex decision problem—how can a high degree of automation be obtained nevertheless?
- How can SMT-based reasoning be combined with model-based reasoning?
- What is the best way to incorporate arithmetic simplification available in computer algebra systems into deductive frameworks?
- How can the specific structure of arithmetic problems generated by deduction systems be exploited?

In addition to the technical contributions, the seminar participants attempted in an open discussion session to identify the major trends and open questions around Deduction and Arithmetic. The outcome of this discussion is summarized in section 3.

2 Table of Contents

3 Trends and Open Questions in Deduction and Arithmetic

All seminar participants; notes taken by Tjark Weber, edited and completed by Reiner Hähnle

A major trend is the growing interest in and partial support of *non-linear arithmetic*, including not only multiplication, but even transcendental functions, and ranging over floating-point data types.

Support for non-linear arithmetic and floating-point data types is particularly important for verification of embedded systems and hybrid systems.

Current tool support for non-linear arithmetic and floating-point data types is immature for the time being, but the community expects considerable progress and activity in this area.

At the same time, the problems around *linear arithmetic with quantifiers* are far from being solved satisfactorily either and work on this front continues as well.

Model-based and *model-guided* approaches are prominent and are perceived as crucial for progress, in particular, to help tackling the partially open problem of how to handle *quantifiers*.

Formal specification of code that uses floating-point types is essentially an unsolved problem. At the moment, generic precision assertions using intervals or deltas are employed, but how to come up with functional specifications is unclear. It was widely agreed that is in general infeasible to write specifications completely by hand, they should be generated at least semi-automatically. The specification problem is compounded by the lack the of semantic abstractions that would make specifications interchangeable.

It was agreed that currently there is a *certification gap* in the tool chain when SMT solvers are involved due to the lack of a common proof format for them. It is desirable to continue work on a common proof framework for SMT solvers, such as the Logical Framework with Side Conditions (LFSC).

4 Overview of Talks

4.1 Approximate decidability and verification of hybrid systems

Jeremy Avigad (Carnegie Mellon University, US)

Joint work of Sicun Gao; Avigad, Jeremy; Clarke, Edmund
Main reference S. Gao, J. Avigad, E. M. Clarke, "Delta-complete decision procedures for satisfiability over the reals," in Proc. of the 27th Annual IEEE Symposium on Logic in Computer Science (LICS'12), pp. 305–314, IEEE, 2012.
URL http://dx.doi.org/10.1109/LICS.2012.41

Control systems that combine analog and discrete components are now ubiquitous. Methods of verifying discrete components are by now well understood, but the task of modeling an analog component, whose evolution over time is often described as the solution to a system of differential equations, poses new challenges. In particular, issues of decidability and complexity limit the reach of what can be done purely symbolically. I will discuss a general framework, based on the methods of computable analysis, for reasoning about specifications using numerical approximations. I will discuss the notion of a "δ-decision procedure", which can be used to evaluate a verification claim. A positive answer provides a guarantee that the system description meets its specification. A negative answer indicates that either the system is unsafe, or that a small perturbation would render it so. I will show that the set of first-order bounded sentences involving computable functions on the real numbers is delta-decidable. I will also briefly discuss complexity considerations, and an implementation, "dreal", due to Gao, Soonho Kong, and Clarke.

References

1 Sicun Gao, Jeremy Avigad, and Edmund Clarke.,"Delta-decidability over the reals." In *Logic in Computer Science (LICS)*, 305–314, 2012.

2 Sicun Gao, Jeremy Avigad, and Edmund Clarke, "Delta-complete decision procedures for satisfiability over the reals." In Bernard Gramlich et al., eds., *International Joint Conference on Automated Reasoning (IJCAR)*, 286–300, 2012.

3 Sicun Gao, Soonho Kong, and Edmund Clarke, "dReal: An SMT Solver for Nonlinear Theories of Reals." In *International Conference on Automated Deduction (CADE)*, 2013.

4.2 Hierarchic superposition with weak abstraction and the Beagle theorem prover

Peter Baumgartner (NICTA, Canberra, AU)

Joint work of Baumgartner, Peter; Waldmann, Uwe
Main reference P. Baumgartner, U. Waldmann, "Hierarchic superposition with weak abstraction," in Proc. of the 24th Int'l Conf. on Automated Deduction (CADE'13), LNCS, Vol. 7898, pp. 39–57, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-38574-2_3

Many applications of automated deduction require reasoning in first-order logic modulo background theories, in particular some form of integer arithmetic. A major unsolved research challenge is to design theorem provers that are "reasonably complete" even in the presence of free function symbols ranging into a background theory sort. The earlier hierarchic superposition calculus of Bachmair, Ganzinger, and Waldmann [1] already supports such symbols, but, not optimally. We have devised a new calculus, hierarchic superposition with weak abstraction, which rectifies this situation by introducing a novel form of clause abstraction, a core component in the hierarchic superposition calculus for transforming clauses into a form needed for internal operation [2]. Additionally, it includes a definition rule that is generally useful to find refutations more often, and, specifically, gives completeness for the clause logic fragment where all background-sorted terms are ground.

The talk provides an overview of the calculus, its implementation in the Beagle theorem prover and experiments with it.

References

1 Leo Bachmair, Harald Ganzinger, and Uwe Waldmann. Refutational theorem proving for hierarchic first-order theories. *Appl. Algebra Eng. Commun. Comput*, 5:193–212, 1994.

2 Peter Baumgartner and Uwe Waldmann. Hierarchic superposition with weak abstraction. In Maria Paola Bonacina, editor, *CADE-24 – The 24th International Conference on Automated Deduction*, volume 7898 of *Lecture Notes in Artificial Intelligence*, pages 39–57. Springer, 2013.

4.3 Basic Hilbert basis

Nikolaj Bjørner (Microsoft Research, Redmond, US)

Joint work of Bjørner, Nikolaj; Voronkov, Andrei

We present a new algorithm for computing Hilbert bases and present an efficient implementation using tailored data-structures. A profound feature of the algorithm is that it is also a constructive proof that *every* basis vector of a Hilbert Basis requires at a polynomial number of bits to represent. This re-establishes that Integer Linear Programming is in NP, and it shows that all minimal basis solutions are small.

4.4 Arithmetic in Sledgehammer

Jasmin Christian Blanchette (Technische Universität München, DE)

Joint work of Blanchette, Jasmin Christian; Böhme, Sascha; Paulson, Lawrence C.
Main reference J. C. Blanchette, S. Böhme, L. C. Paulson, "Extending Sledgehammer with SMT Solvers", J. Autom. Reasoning 51(1):109–128, 2013.
URL http://dx.doi.org/10.1007/s10817-013-9278-5

Sledgehammer is a subsystem of Isabelle/HOL that integrates automatic theorem provers (ATPs). It uses, among others, the first-order resolution provers E, SPASS, Vampire and the SMT solvers CVC3, Yices, and Z3 as backends. Interpreted arithmetic is used for the SMT solvers, but the impact is not as great as one might have expected. This talk attempted to give a few explanations and suggested directions for future research.

4.5 Sound compilation of reals

Eva Darulova (EPFL, Lausanne, CH)

Joint work of Darulova, Eva; Kuncak, Viktor
Main reference E. Darulova, V. Kuncak, "Sound Compilation of Reals," in Proc. of the 41st Annual ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages (POPL'14), pp. 235–248, ACM, 2014.
URL http://dx.doi.org/10.1145/2535838.2535874

Writing accurate numerical software is hard because of many sources of unavoidable uncertainties, including finite numerical precision of implementations. We present a programming model where the user writes a program in a real-valued implementation and specification language that explicitly includes different types of uncertainties. We then present a compilation algorithm that generates a conventional implementation that is guaranteed to meet the desired precision with respect to real numbers. Our verification step generates verification conditions that treat different uncertainties in a unified way and encode reasoning about floating-point roundoff errors into reasoning about real numbers. We show that current state-of-the art SMT solvers do not scale well to solving such verification conditions. We propose a new procedure that combines exact SMT solving over reals with approximate and sound affine and interval arithmetic. We report on results from our initial prototype implementation.

4.6 Making invariants inductive

Stephan Falke (Karlsruhe Institute of Technology, DE)

Joint work of Falke, Stephan; Kapur, Deepak

While non-inductive invariants might be easier to write, inductive invariants are easier to prove. In this talk, I will present an algorithm that aims at strengthening a given potential invariant so it becomes inductive. I will discuss properties of the algorithm and pose open questions.

4.7 The poor man's way to integrate non-linear real arithmetic reasoning capabilities within SMT

Pascal Fontaine (LORIA, Nancy, FR)

Joint work of Déharbe, David; Fontaine, Pascal; Sturm, Thomas

In this talk, we report some very preliminary results on a cooperation between the veriT SMT solver and Redlog. Thanks to this cooperation, the language handled by veriT now accepts non-linear real arithmetic expressions. The technique is based on a model-based combination. The linear arithmetic and congruence closure decision procedures are used as simplifiers for the set of constraints given to Redlog.

4.8 Deduction and arithmetic – a functional marriage: iSAT, odeSAT, and related procedures

Martin Fränzle (Universität Oldenburg, DE)

Joint work of Fränzle, Martin; Eggers, Andreas; Herde, Christian; Teige, Tino; Becker, Bernd; Kupferschmid, Stefan; Scheibler, Karsten; Schubert, Tobias; Ratschan, Stefan; Ramdani, Nacim; Nedialkov, Nedlialko

Over the last decade, arithmetic SAT modulo theory (SMT) solving has found wide-spread application within diverse analysis tasks for hybrid discrete-continuous systems, cyber-physical systems, or software systems involving numerical computations. Traditional SMT approaches are, however, confined to decidable theories, i.e. small fragments of arithmetic. Combining ideas from abstract interpretation, namely to manipulate lattices of subsets of the value domain, with ideas from verified computer arithmetic, namely to use safe numeric interval enclosures, and with techniques from SMT solving, namely conflict-driven clause learning, this restriction can be relaxed in practice. While the resulting procedures, which represent a "functional marriage" between automated deduction and computer arithmetic, are necessarily incomplete, they can solve many intricate problems in practice: For complex-structured Boolean combinations of arithmetic constraints involving transcendental functions and thousands of variables, or for combinations of arithmetic constraints involving first-order relations defined by ordinary differential equations, they are able to rigorously prove unsatisfiability, do provide strong hints (and sometimes even proofs) for satisfiability, and can compute Craig interpolants.

References

1 Martin Fränzle, Christian Herde, Stefan Ratschan, Tobias Schubert, and Tino Teige. Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure. *Journal on Satisfiability, Boolean Modeling and Computation – Special Issue on SAT/CP Integration*, 1:209- -236, 2007.

2 Andreas Eggers, Martin Fränzle, and Christian Herde. SAT modulo ODE: A direct SAT approach to hybrid systems. In Sungdeok (Steve) Cha, Jin-Young Choi, Moonzoo Kim, Insup Lee, and Mahesh Viswanathan, editors, *Proceedings of the 6th International Symposium on Automated Technology for Verification and Analysis (ATVA'08)*, volume 5311 of Lecture Notes in Computer Science (LNCS), pages 171–185. Springer-Verlag, 2008.

4.9 Cooperation for better termination proving

Carsten Fuhs (University College London, GB)

Joint work of Brockschmidt, Marc; Cook, Byron; Fuhs, Carsten
Main reference M. Brockschmidt, B. Cook, C. Fuhs, "Better termination proving through cooperation," in Proc. of 25th Int'l Conf. on Computer Aided Verification (CAV'13), LNCS, Vol. 8044, pp. 413–429, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-39799-8_28

One of the difficulties of proving program termination is managing the subtle interplay between the finding of a termination argument and the finding of the argument's supporting invariant. We propose a new mechanism that facilitates better cooperation between these two types of reasoning. In an experimental evaluation we find that our new method leads to dramatic performance improvements.

References

1 Marc Brockschmidt, Byron Cook, and Carsten Fuhs. *Better termination proving through cooperation.* In *Proc. CAV '13*, volume 8044 of *LNCS*, pages 413–429, 2013.

4.10 Alternating runtime and size complexity analysis of integer programs

Jürgen Giesl (RWTH Aachen, DE)

We present a new modular approach to automatic complexity analysis. Based on a novel alternation between finding symbolic time bounds for program parts and using these to infer size bounds on program variables, we can restrict each analysis step to a small part of the program while maintaining a high level of precision. Extensive experiments with the implementation of our method demonstrate its performance and power in comparison with other tools. In particular, our method finds bounds for many programs whose complexity could not be analyzed by automatic tools before.

4.11 Simple interpolation for floating-point arithmetic with abstract CDCL

Alberto Griggio (Bruno Kessler Foundation, Trento, IT)

Joint work of Brain, Martin; D'Silva, Vijay; Griggio, Alberto; Haller, Leopold; Kroening, Daniel
Main reference M. Brain, V. D'Silva, A. Griggio, L. Haller, D. Kroening, "Interpolation-Based Verification of Floating-Point Programs with Abstract CDCL," in Proc. of the 20th Int'l Symp. on Static Analysis (SAS'13), LNCS, Vol. 7935, pp. 412–432, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-38856-9_22

One approach for SMT solvers to improve efficiency is to delegate reasoning to abstract domains. Solvers using abstract domains do not support interpolation and cannot be used for interpolation-based verification. We extend Abstract Conflict Driven Clause Learning (ACDCL) solvers with proof generation and interpolation. Our results lead to the first interpolation procedure for floating-point logic and subsequently, the first interpolation-based verifiers for programs with floating-point variables. We demonstrate the potential of this approach by verifying a number of programs which are challenging for current verification tools.

4.12 Model-constructing satisfiability calculus

Dejan Jovanović (SRI, Menlo Park, US)

Joint work of Jovanović, Dejan; Barrett, Clark; de Moura, Leonardo
Main reference D. Jovanović, C. Barrett, L. De Moura, "Design and implementation of the model-constructing satisfiability calculus." Presentation at 2013 Formal Methods in Computer-Aided Design Conference (FMCAD'13).
URL http://www.cs.utexas.edu/users/hunt/FMCAD/FMCAD13/papers/71-Model-Construction-SAT-Calculus.pdf

We present the Model Constructing Satisfiability (MCSat) calculus. MCSat calculus generalizes ideas found in CDCL-style propositional SAT solvers to SMT solvers, and provides a common framework where recent model-based procedures and techniques can be justified and combined. We describe how to incorporate support for linear real arithmetic and uninterpreted function symbols in the calculus. We report encouraging experimental results, where MCSat performs competitive with the state-of-the art SMT solvers without using pre-processing techniques and ad-hoc optimizations. The implementation is flexible, additional plugins can be easily added, and the code is freely available.

4.13 Practical exploitation of mixed integer programming solvers by SMT solvers

Tim A. King (New York University, US)

Simplex based methods used within SMT solvers become overwhelmed by problems that are efficiently solved by the Simplex solvers used within Mixed Integer Programming solvers.

This talk will describe practical usage of MIP solvers from within SMT solvers to extend SMT solvers to be able to handle such benchmarks. A preliminary implementation of this technique solves the more SMTLIB QFLRA benchmarks then other state-of-art SMT solvers.

4.14 Solving linear arithmetic by bound propagation

Konstantin Korovin (University of Manchester, GB)

Joint work of Korovin, Konstantin; Voronkov, Andrei
Main reference K. Korovin, A. Voronkov, "Solving Systems of Linear Inequalities by Bound Propagation," in Proc. of the 23rd Int'l Conf. on Automated Deduction, LNCS, Vol. 6803, pp. 369–383, Springer, 2011.
URL http://dx.doi.org/10.1007/978-3-642-22438-6_28

Bound propagation is a method for solving systems of linear inequalities using DPLL-style reasoning, shown to be complete in our previous work ([1], extending conflict resolution [2]). The bound propagation method adapts constraint propagation, dynamic variable ordering, lemma learning and backjumping. In this talk I overview the bound propagation method, discuss non-trivial issues such as termination, and present recent implementation and experimental results [3, 4]. Joint work with Ioan Dragan, Laura Kovács, Nestan Tsiskaridze and Andrei Voronkov.

References

1 K. Korovin and A. Voronkov Solving Systems of Linear Inequalities by Bound Propagation. *Proc. of the 23rd International Conference on Automated Deduction, (CADE 2011)*, 369–383, LNAI, vol 6803, Springer, 2011.
2 K. Korovin, N. Tsiskaridze and A. Voronkov. Conflict Resolution. *Proc. of the 15th International Conference on Principles and Practice of Constraint Programming (CP'09)*, 509–523, Lecture Notes in Computer Science, vol 5732, Springer, 2009.
3 K. Korovin, N. Tsiskaridze and A. Voronkov. Implementing Conflict Resolution. *Proc. of the 8th International Conference on Perspectives of Systems Informatics (PSI'2011)*, LNCS, vol 7162, Springer, 2012.
4 I. Dragan, K. Korovin, L. Kovacs and A. Voronkov. Bound Propagation for Arithmetic Reasoning in Vampire. Proc. of SYNASC'13, IEEE, 2013.

4.15 An SMT-based approach to memory access optimization problem

Marek Košta (Max-Planck-Institut für Informatik, Saarbrücken, DE)

Joint work of Košta, Marek; Karrenberg, Ralf; Sturm, Thomas
Main reference R. Karrenberg, M. Košta, T. Sturm, "Presburger Arithmetic in Memory Access Optimization for Data-Parallel Languages," in Proc. of the 9th Int'l Symp. on Frontiers of Combining Systems (FroCoS'13), LNCS, Vol. 8152, pp. 56–70, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-40885-4_5

To exploit capabilities of modern SIMD capable CPUs, state-of-the-art compilers use vectorization techniques. One of the main drawbacks of these techniques is that they serialize memory access operations. We begin with memory access optimization problem definition and describe how its solution can help to generate vectorized, i.e. more efficient code in presence of memory accesses. Then we will refer about our previous work, which uses SMT solving to solve the memory access optimization problem. Next, we will describe our current work: Development of a system using Z3 library capable of fully-automatic solution of the memory access optimization problem. We will conclude with a few theoretical and practical questions motivated by our current work.

4.16 Acceleration for Petri nets

Jérôme Leroux (University of Bordeaux, FR)

Main reference J. Leroux, "Acceleration for Petri Nets," in Proc. of the 11th Int'l Symp. on Automated Technology for Verification and Analysis (ATVA'13), LNCS, Vol. 8172, pp. 1–4, Spinger, 2013.
URL http://dx.doi.org/10.1007/978-3-319-02444-8_1

The reachability problem for Petri nets is a central problem of net theory. The problem is known to be decidable by inductive invariants definable in the Presburger arithmetic. When the reachability set is definable in the Presburger arithmetic, the existence of such an inductive invariant is immediate. However, in this case, the computation of a Presburger formula denoting the reachability set is an open problem. Recently this problem got closed by proving that if the reachability set of a Petri net is definable in the Presburger arithmetic, then the Petri net is flatable, i.e. its reachability set can be obtained by runs labeled by words in a bounded language. As a direct consequence, classical algorithms based on acceleration techniques effectively compute a formula in the Presburger arithmetic denoting the reachability set. In this presentation, the framework of verification of infinite-state systems based on acceleration techniques is recalled. We also explain the completeness of this method on the computation of Presburger formulas denoting the reachability sets of Petri nets.

4.17 Automated (formal) proofs of summation identities

Assia Mahboubi (INRIA Saclay–Île-de-France–Orsay, FR)

Joint work of Mahboubi, Assia; Chyzak, Frédéric; Sibut-Pinote, Thomas; Tassi, Enrico

Among the most cited references in the mathematical literature are the handbooks of properties of special functions or sequences. These dictionaries consist of tables of results about functions arising naturally in applications like physics, number theory, economy etc. It happens that a large subset of these functions belongs to a well-behaved class of objects, which benefits from the theory of so-called holonomic systems. In particular, it is possible to increase dramatically the confidence in the results displayed in the handbooks by relying on algorithmic proofs and computer algebra systems. In this talk I will discuss the issues raised by the certification of these theorems by a proof assistant.

4.18 From ordinal interpretations to elementary interpretations

Aart Middeldorp (Universität Innsbruck, AT)

Joint work of Winkler, Sarah; Zankl, Harald; Middeldorp, Aart
Main reference S. Winkler, H. Zankl, A. Middeldorp, "Beyond Peano Arithmetic – Automatically Proving Termination of the Goodstein Sequence," in Proc. of the 24th Int'l Conf. on Rewriting Techniques and Applications (RTA'13), LIPIcs, Vol. 21, pp. 335–351, Dagstuhl Publishing, 2013.
URL http://dx.doi.org/10.4230/LIPIcs.RTA.2013.335

Kirby and Paris (1982) proved in a celebrated paper that a theorem of Goodstein (1944) cannot be established in Peano (1889) arithmetic. In a recent paper we presented an encoding of Goodstein's theorem as a termination problem of a finite rewrite system. Using a novel implementation of ordinal interpretations, we were able to automatically prove termination of this system, resulting in the first automatic termination proof for a system whose derivational complexity is not multiply recursive. After recapitulating this work, we discuss the challenges when trying to implement the elementary interpretations of Lescanne (1995).

4.19 Death by a thousand cuts (worst-case execution time by bounded model checking)

David Monniaux (VERIMAG, Grenoble, FR)

Joint work of Monniaux, David; Henry, Julien; Maïza, Claire; Caminha, Diego

The trace semantics of programs (with finite loop unrolling) can be compiled into first-order logic formulas, and thus bounded model checking maps to satisfiability testing for such formulas. Modern satisfiability modulo theory (SMT) solvers currently solve many large instances. Unfortunately, if certain non-functional properties such as timing and energy are naively encoded into the formula, the SMT problems become intractable since they contain "diamond formulas". We propose a general redundant encoding scheme for such properties, resulting in tractable SMT problems: we conjoin to the original problem some "cuts", which do not change the solution set but kill the complexity of the solving.

We illustrate this encoding with worst-case execution time (WCET) of loop-free programs. In real-time systems, it is often necessary to bound the WCET of program fragments; for instance, in safety-critical control systems, this time should be less than the period of the control loop. The conventional approach is to first run special static analyses which give for each basic block an upper bound for its WCET, taking into account information from the history of the computation (e.g. possible pipeline and cache states), then reassemble these local WCET into a global WCET through a path analysis. Unfortunately, this path analysis may take into account paths that are infeasible due to the semantics of the instructions, resulting in over-estimation of the WCET. We replace this path analysis by optimization modulo theory.

We experimented our analysis on benchmark and industrial programs, gaining as much as 53% improvement on the bound on WCET for a fly-by-wire program. On most of these programs, a naive encoding results in timeout, but with our encoding the analysis terminates within minutes.

4.20 Non-numerical permissions for concurrent reasoning

Wojciech Mostowski (University of Twente, NL)

We present a new, symbolic system for permission accounting that can be easily used with first-order logic based reasoning systems typically used in program verification. Permission accounting is a core building block in approaches to modular, thread-local reasoning about concurrent programs. Our permission system differs from the existing ones in that it provides a symbolic and global view of permissions, rather than value-based and thread-local one. That in turn enables (a) better understanding of permission tracking from the point of the view of the specifier, (b) specification of complex permission transfer scenarios, and (c) more efficient reasoning for the verification tools (in particular, no reasoning about rational numbers is required). Our system is based on the idea of "I-owe-you" chains of permission owners to track the history of permission transfers, and the idea of symbolic permission slicing to divide permissions between multiple owners/threads. Underneath, special lists with dedicated operations are used. We axiomatised our permission system and its vital properties in the KeY verifier as well as in PVS. KeY is a verification tool for Java programs based on first-order dynamic logic and our primary target to employ our permission system for reasoning about multi-threaded Java programs. Initial results with the verification of actual Java programs using our permission system and KeY are also discussed.

4.21 Quantifier elimination for dense and discrete linear orders

Tobias Nipkow (Technische Universität München, DE)

In earlier work I verified a number of quantifier elimination procedures for dense linear orders without endpoints in Isabelle [1]. In ongoing work I have extend these quantifier elimination procedures to discrete orderings and to orderings with endpoints (typically plus and minus infinity). It is known from the literature that these theories admit quantifier elimination, but my focus is on efficient procedures. It turns out that the method of interior points (some arbitrary point between a lower and upper bound, e.g. $(l + u)/2$) and of infinitesimals (due to Loos and Weispfenning) can be extended to endpoints and to discrete orders.

References

1 Tobias Nipkow. *Linear Quantifier Elimination.* In A. Armando, P. Baumgartner, and G. Dowek (eds.), *International Joint Conference on Automated Reasoning (IJCAR)*, LNCS 5195, pages 18–33, Springer, 2008.

4.22 Exact global optimization on demand

Grant Olney Passmore (University of Edinburgh, GB)

Joint work of Passmore, Grant Olney; de Moura, Leonardo

We present a method for exact global nonlinear optimization based on a real algebraic adaptation of the conflict-driven clause learning (CDCL) approach of modern SAT solving. This method allows polynomial objective functions to be constrained by real algebraic constraint systems with arbitrary boolean structure. Moreover, it can correctly determine when an objective function is unbounded, and can compute exact infima and suprema when they exist. The method requires computations over real closed fields containing infinitesimals (cf. [1]).

References

1 Leonardo de Moura and Grant Olney Passmore. *Computation in Real Closed Infinitesimal and Transcendental Extensions of the Rationals.* In Proceedings of the 24th International Conference on Automated Deduction (CADE) (2013)

4.23 Automating separation logic using SMT

Ruzica Piskac (Yale University, US)

Joint work of Piskac, Ruzica; Wies, Thomas; Zufferey, Damien
Main reference R. Piskac, T. Wies, D. Zufferey, "Automating Separation Logic Using SMT," in Proc. of the . 25th Int'l Conf. on Computer Aided Verification (CAV'13), LNCS, Vol. 8044, pp. 773–789, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-39799-8_54

Separation logic (SL) has gained widespread popularity because of its ability to succinctly express complex invariants of a program's heap configurations. Several specialized provers have been developed for decidable SL fragments. However, these provers cannot be easily extended or combined with solvers for other theories that are important in program verification, e.g., linear arithmetic. In this talk, we present a reduction of decidable SL fragments to a decidable first-order theory that fits well into the satisfiability modulo theories (SMT) framework. We show how to use this reduction to automate satisfiability, entailment, frame inference, and abduction problems for separation logic using SMT solvers. Our approach provides a simple method of integrating separation logic into existing verification tools that provide SMT backends, and an elegant way of combining SL fragments with other decidable first-order theories.

4.24 Logic of hybrid games

André Platzer (Carnegie Mellon University, US)

Main reference A. Platzer, "A Complete Axiomatization of Differential Game Logic for Hybrid Games," School of Computer Science, Carnegie Mellon University, CMU-CS-13-100R, January 2013, extended in revised version from July 2013.
URL http://reports-archive.adm.cs.cmu.edu/anon/2013/CMU-CS-13-100R.pdf

Hybrid systems model cyber-physical systems as dynamical systems with interacting discrete transitions and continuous evolutions along differential equations. They arise frequently in many application domains, including aviation, automotive, railway, and robotics. This talk studies hybrid games, i.e. games on hybrid systems combining discrete and continuous dynamics. Unlike hybrid systems, hybrid games allow choices in the system dynamics to be resolved adversarially by different players with different objectives.

This talk describes how logic and formal verification can be lifted to hybrid games [2, 1]. The talk describes a logic for hybrid systems called differential game logic dGL. The logic dGL can be used to study the existence of winning strategies for hybrid games, i.e. ways of resolving the player's choices in some way so that he wins by achieving his objective for all choices of the opponent. Hybrid games are determined, i.e. one player has a winning strategy from each state, yet their winning regions may require transfinite closure ordinals. The logic dGL, nevertheless, has a sound and complete axiomatization relative to any expressive logic. Separating axioms are identified that distinguish hybrid games from hybrid systems. Finally, dGL is proved to be strictly more expressive than the corresponding logic of hybrid systems.

References

1 André Platzer. *A Complete Axiomatization of Differential Game Logic for Hybrid Games.* School of Computer Science, Carnegie Mellon University, CMU-CS-13-100, January 2013, extended in revised version from July 2013.
2 André Platzer. *Differential Game Logic for Hybrid Games.* School of Computer Science, Carnegie Mellon University, CMU-CS-12-105, March 2012.

4.25 Proving unsatisfiability in non-linear arithmetic by duality

Enric Rodríguez-Carbonell (UPC, BarcelonaTech, ES)

Joint work of Larraz, Daniel; Oliveras, Albert; Rodríguez-Carbonell, Enric; Rubio, Albert

Non-linear problems arise in many contexts, for instance, in the generation of invariants and ranking functions following the constraint-based program analysis approach. Solvers based on linearization by case analysis efficiently find solutions for satisfiable instances. On the other hand, unsatisfiability is difficult to prove with this method. In this talk we propose to prove a conjunction of non-linear atoms to be unsatisfiable by finding a Positivstellensatz refutation certificate, which is more amenable to satisfiability-goaled non-linear solvers.

4.26 Exploring interpolants

Philipp Rümmer (Uppsala University, SE)

Joint work of Rümmer, Philipp; Subotić, Pavle
Main reference Philipp Rümmer, Pavle Subotić. Exploring Interpolants. FMCAD 2013.

Craig Interpolation is a standard method to construct and refine abstractions in model checking. To obtain abstractions that are suitable for the verification of software programs or hardware designs, model checkers rely on theorem provers to find the right interpolants, or interpolants containing the right predicates, in a generally infinite lattice of interpolants for any given interpolation problem. We present a semantic and solver-independent framework for systematically exploring interpolant lattices, based on the notion of interpolation abstraction [1]. We discuss how interpolation abstractions can be constructed for a variety of logics, and how they can be exploited in the context of software model checking.

References

1 Philipp Rümmer, Pavle Subotić. *Exploring Interpolants.* Formal Methods in Computer-Aided Design (FMCAD). Portland, USA, 2013.

4.27 Some challenges in applied software bounded model checking

Carsten Sinz (Karlsruhe Institute of Technology, DE)

Joint work of Sinz, Carsten; Falke, Stephan; Merz, Florian

In this talk I will give a personal view on what challenges software analysis tools (like bounded model checkers) are currently facing, especially when applied to low-level system and control software. The intention of the talk is not so much to present finished work, but to discuss ideas for future research directions.

4.28 Hierarchical reasoning and model generation in the verification of hybrid systems

Viorica Sofronie-Stokkermans (Universität Koblenz-Landau, DE)

Main reference V. Sofronie-Stokkermans, "Hierarchical Reasoning and Model Generation for the Verification of Parametric Hybrid Systems," in Proc. of the 24th Int'l Conf. on Automated Deduction (CADE'13), LNCS, Vol. 7898, pp. 360–376, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-38574-2_25

We study possibilities of using hierarchical reasoning, quantifier elimination and model generation for the verification of parametric hybrid systems, where the parameters can be constants or functions. Our goal is to automatically provide guarantees that such systems satisfy certain safety or invariance conditions.

We first analyze the possibility of automatically generating such guarantees in the form of constraints on parameters, then show that we can also synthesise so-called criticality functions, typically used for proving stability and/or safety of hybrid systems.

We illustrate our methods on several examples. The results are presented in detail in [1].

References

1 Viorica Sofronie-Stokkermans. *Hierarchical Reasoning and Model Generation for the Verification of Parametric Hybrid Systems.* In Maria Paola Bonacida (ed.) Automated Deduction – CADE-24 – 24th International Conference on Automated Deduction, LNCS 7898, pages 360–376, Springer 2013.

4.29 Certified abstract completion

Christian Sternagel (JAIST, Nomi, JP)

The textbook proof of soundness of the Knuth-Bendix completion procedure is rather involved. To make this important result more accessible, especially in a classroom situation, Hirokawa and Middeldorp recently found an alternative proof via peak-decreasingness (which is a weaker but also simpler variant of the decreasing diagrams method). I present an Isabelle/HOL formalization of this new proof and compare it to a formalization of the traditional proof which is part of IsaFoR (a library of formalized results about rewriting).

4.30 Partial certification for termination proofs and more

Rene Thiemann (Universität Innsbruck, AT)

Since untrusted termination tools will always use techniques that have not been formally verified in some proof assistant, it is hard to fully certify the generated proofs. We present an approach which still allows to certify large parts of the proof, where we support both an online and an offline modus for certification.

We further report on initial steps towards the certification of complexity proofs where we focus on problems with respect to arithmetic.

4.31 Integrating SAT, QBF and SMT solvers with interactive theorem provers

Tjark Weber (Uppsala University, SE)

This talk describes integrations of various automated solvers (for SAT, QBF and SMT) with the interactive theorem provers Isabelle/HOL and HOL4. Our integrations demonstrate that LCF-style proof checking is feasible for those solvers. We thereby increase not only automation in interactive theorem proving, but also confidence in the correctness of solver results.

5 Program

Monday, 7 October 2013

09:00–09:30	Organizers Welcome, Announcements, Introduction of Participants
09:30–10:00	Jürgen Giesl Analyzing Runtime and Size Complexity of Integer Programs
10:00–10:30	André Platzer Logic of Hybrid Games
11:00–11:30	Peter Baumgartner Hierarchic Superposition With Weak Abstraction and the Beagle Theorem Prover
11:30–12:00	Wojciech Mostowski Non-numerical Permissions for Concurrent Reasoning
14:00–14:30	Enric Rodríguez-Carbonell Proving Unsatisfiability in Non-Linear Arithmetic by Duality
14:30–15:00	Wolfgang Ahrendt Verifying (In)stability in Floating-point Programs by Increasing Precision, using SMT Solving
15:00–15:30	Carsten Fuhs Cooperation for Better Termination Proving
16:00–16:30	Marek Košta An SMT-Based Approach To Memory Access Optimization Problems
16:30–17:00	Christian Sternagel Certified Abstract Completion
17:00–17:30	David Monniaux Death by a Thousand Cuts

Tuesday, 8 October 2013

09:00–09:30	Aart Middeldorp
	From Ordinal Interpretations to Elementary Interpretations
09:30–10:00	Pascal Fontaine
	The Poor Man's Way to Integrate Non-Linear Real Arithmetic Reasoning Capabilities within SMT
10:00–10:30	Jeremy Avigad
	Approximate Decidability and the Verification of Hybrid Systems
11:00–11:30	Viorica Sofronie-Stokkermans
	Hierarchical Reasoning and Model Generation in Verification
11:30–12:00	Eva Darulova
	Sound Compilation of Reals
14:00–14:30	Assia Mahboubi
	Automated (Formal) Proofs of Summation Identities
14:30–15:00	Konstantin Korovin
	Solving Linear Arithmetic by Bound Propagation
15:00–15:30	Arie Gurfinkel
	Exploring Interpolants
16:00–16:30	Bernhard Beckert
	On the Specification and Verification of Voting Schemes
16:30–17:00	Jérôme Leroux
	Computing Vector Addition System Reachability Sets
17:00–17:30	Philipp Rümmer
	Interpolation for Software Verification

Wednesday, 9 October 2013

09:00–09:30	Christoph Weidenbach
	Hierarchic Superposition – Current Status and Future Steps
09:30–10:00	Dejan Jovanović
	Model-Constructing Satisfiability Calculus
10:00–10:30	Nikolaj Bjørner
	Basic Hilbert Basis
11:00–11:30	Tjark Weber
	Integrating SAT, QBF and SMT Solvers with Interactive Theorem Provers
11:30–12:00	Jasmin Christian Blanchette
	Arithmetic in Sledgehammer

Thursday, 10 October 2013

09:00–09:30	Leonardo de Moura Arithmetic Procedures in the Model-Constructing Satisfiability Calculus
09:30–10:00	Tobias Nipkow Linear Quantifier Elimination for Linear Orderings with Endpoints
10:00–10:30	Thomas Sturm Satisfiability Checking for the Sciences
11:00–11:30	Stephan Falke Making Invariants Inductive
11:30–12:00	Grant Passmore Exact Global Optimization on Demand
14:00–14:30	Ruzica Piskac Automating Separation Logic Using SMT
14:30–15:00	Carsten Sinz Some Challenges in Applied Software Bounded Model Checking
16:00–16:30	Tim King Floating Point Simplex Solvers for SMT
16:30–17:00	Alberto Griggio Simple Interpolation for Floating-Point Arithmetic with Abstract CDCL

Friday, 10 October 2013

09:00–09:30	Martin Fränzle Deduction and Arithmetic – a Functional Marriage (iSAT, odeSAT, and Related Procedures)
09:30–10:00	René Thiemann Partial Certification for Termination Proofs and More
10:00–10:30	James J. Hunn Realtime Java and Formal Methods: Use and Open Issues
11:00–12:00	Organizers Wrap-Up Session

Participants

- Wolfgang Ahrendt
Chalmers UT – Göteborg, SE
- Jeremy Avigad
Carnegie Mellon University, US
- Peter Baumgartner
NICTA – Canberra, AU
- Bernhard Beckert
KIT – Karlsruhe Institute of Technology, DE
- Nikolaj Bjørner
Microsoft Res. – Redmond, US
- Jasmin Christian Blanchette
TU München, DE
- Richard Bubel
TU Darmstadt, DE
- Eva Darulova
EPFL – Lausanne, CH
- Leonardo de Moura
Microsoft Res. – Redmond, US
- Stephan Falke
KIT – Karlsruhe Institute of Technology, DE
- Pascal Fontaine
LORIA – Nancy, FR
- Martin Fränzle
Universität Oldenburg, DE
- Carsten Fuhs
University College London, GB
- Jürgen Giesl
RWTH Aachen, DE
- Alberto Griggio
Bruno Kessler Foundation – Trento, IT
- Arie Gurfinkel
Carnegie Mellon University, US
- Reiner Hähnle
TU Darmstadt, DE
- James J. Hunt
aicas GmbH – Karlsruhe, DE
- Dejan Jovanovic
SRI – Menlo Park, US
- Tim A. King
New York University, US
- Konstantin Korovin
University of Manchester, GB
- Marek Kosta
MPI für Informatik – Saarbrücken, DE
- Jerome Leroux
University of Bordeaux, FR
- Assia Mahboubi
INRIA Saclay – Île-de-France – Orsay, FR
- Aart Middeldorp
Universität Innsbruck, AT
- David Monniaux
VERIMAG – Grenoble, FR
- Wojciech Mostowski
University of Twente, NL
- Tobias Nipkow
TU München, DE
- Grant Olney Passmore
University of Cambridge & University of Edinburgh, GB
- Ruzica Piskac
Yale University, US
- André Platzer
Carnegie Mellon University, US
- Enric Rodríguez-Carbonell
UPC – BarcelonaTech, ES
- Philipp Rümmer
Uppsala University, SE
- Peter H. Schmitt
KIT – Karlsruhe Institute of Technology, DE
- Carsten Sinz
KIT – Karlsruhe Institute of Technology, DE
- Viorica Sofronie-Stokkermans
Universität Koblenz-Landau, DE
- Christian Sternagel
JAIST – Nomi, JP
- Thomas Sturm
MPI für Informatik – Saarbrücken, DE
- René Thiemann
Universität Innsbruck, AT
- Cesare Tinelli
Univ. of Iowa – Iowa City, US
- Tjark Weber
Uppsala University, SE
- Christoph Weidenbach
MPI für Informatik – Saarbrücken, DE